3.7）消费端的ack

一:消费端的ack模式

消费端的ack类型:自动ack 和手动ack

做消息限流的时候，我们需要关闭自动ack 然后进行手动ack的确认,若我们业务出现了问题，我们就可以进行nack

重回队列

当消费端进行了nack的操作的时候，我们可以通过设置来进行对消息的重回队列的操作(但是一般我们不会设置重回队列的操作)

代码演示(手动ack 以及重回队列操作)

```
================================================================================
public class producter {
public static void main(String[] args) throws IOException, TimeoutException {
 ConnectionFactory connectionFactory = new ConnectionFactory();
 connectionFactory.setVirtualHost("/");
 connectionFactory.setHost("47.104.128.12");
 connectionFactory.setPort(5672);
Connection connection = connectionFactory.newConnection();
Channel channel = connection.createChannel();

for(int i=0;i<5;i++) {
 Map<String,Object> header = new HashMap<>();
 header.put("num",i+1);
 //设置消息属性
 AMQP.BasicProperties basicProperties = new AMQP.BasicProperties().builder()
 .contentEncoding("utf-8")
 .contentType("application/json")
 .deliveryMode(2).headers(header)
 .build();
 channel.basicPublish("test.ack.exchange","test.ack.key",false,basicProperties,("自定ack消息"+(i+1)).getBytes());
 }
}

================================================================================消
package com.hnnd.mq.NackAndReQueue;

import com.rabbitmq.client.Channel;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.ConnectionFactory;

import java.io.IOException;
import java.util.concurrent.TimeoutException;

/**
 * Created by Administrator on 2018/10/19.
 */
public class Consumer {

    public static void main(String[] args) throws IOException, TimeoutException {
        ConnectionFactory connectionFactory = new ConnectionFactory();
```

```
        connectionFactory.setVirtualHost("/");
        connectionFactory.setHost("47.104.128.12");
        connectionFactory.setPort(5672);

        Connection connection = connectionFactory.newConnection();

        Channel channel = connection.createChannel();

        channel.exchangeDeclare("test.ack.exchange","direct",true,true,false,null);
        channel.queueDeclare("test.ack.queue",true,false,true,null);
        channel.queueBind("test.ack.queue","test.ack.exchange","test.ack.key");

        //gloabl设置为ture 那么就是channel级别的限流，若为false 就是consumer级别的限制流量
        //channel.basicQos(0,1,false);

        //关闭自动签收
        channel.basicConsume("test.ack.queue",false,new AngleCustomConsumer(channel));

    }
}



==========================================================================自定
package com.hnnd.mq.NackAndReQueue;

import com.rabbitmq.client.AMQP;
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.DefaultConsumer;
import com.rabbitmq.client.Envelope;
import org.springframework.util.StringUtils;

import java.io.IOException;

/**
 * Created by Administrator on 2018/10/19.
 */
public class AngleCustomConsumer extends DefaultConsumer {

    private Channel channel;
    /**
     * Constructs a new instance and records its association to the passed-in channel.
     *
     * @param channel the channel to which this consumer is attached
     */
    public AngleCustomConsumer(Channel channel) {
        super(channel);
        this.channel = channel;
    }

    /**
     * 处理消息
     * @param consumerTag
     * @param envelope
     * @param properties
     * @param body
     * @throws IOException
     */
    public void handleDelivery(String consumerTag,Envelope envelope,AMQP.BasicProperties properties, byte[] body)
            throws IOException
    {
        try {
```

```
            Thread.currentThread().sleep(1000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println(properties.getHeaders());
        String num = properties.getHeaders().get("num").toString();
        if(num .equals("1")) {
            System.out.println("业务系统处理消息异常消息重新回队列"+ new String(body));
            channel.basicNack(envelope.getDeliveryTag(),false,true);

        }else {
            System.out.println("自定义的消息消费端");
            System.out.println("consumerTag="+consumerTag);
            System.out.println("envelope="+envelope);
            System.out.println("properties="+properties);
            System.out.println("body="+new String(body));

            //消费端的手动签收,加入我关闭手动签收，也关闭自动签收，那么消费端只会接收到一条消息
            channel.basicAck(envelope.getDeliveryTag(),false);
        }
    }
}
```